



Effectively Handling XML data in bulk



Author	Mamata Das
Date of Creation	24 th October 2008
Email id	mamata.das@wipro.com
Practice	BTS
Reviewed by	Muralidhara Rao Patri, Anant Narayan, Varma S G M





Table of Contents

[1. Introduction.....4](#)

[2. Problem – Chunking big volumes of XML data files into smaller files.....4](#)

[3. Solution.....4](#)

[4. How XML Splitter works.....5](#)

[5. Benefits and Constraints.....6](#)



1.Introduction

Applications dealing with XML data files as datasource have been found to encounter performance issues with big volumes of XML data files. Application components attempting to read large volume of XML data files in one shot can throw unwanted errors and exhaust memory for the other components to run.

A similar situation encountered in one of the EAI projects prompted the delivery team to come up with an effective solution where performance issues were eradicated by using a set of re-usable components that could split a big volume of XML file into smaller and manageable XML data files.

This solution consists of a re-usable library that can be used across EAI and non EAI applications that use XML data files in bulk volume as datasource.

2.Problem – Chunking big volumes of XML data files into smaller files

For plain ASCII files (delimited or fixed length), data reading in chunks of characters or lines is possible since files conform to a recurring pre-defined format or layout.

For XML files however, these solutions cannot be implemented since files conform to well-defined tags, their layouts and their specifications.

XML data files conform to a well-defined XSD or DTD. The files use a pre-defined set of tags according to the layout stated in XSD or DTD. Splitting XML files of large sizes into multiple files conforming to the XML structure is a challenge.

3.Solution

XML files can be parsed using DOM or SAX methodology of parsing.

DOM methodology validates and parses an XML file and builds the XML structure in memory in the form of a tree. DOM parsing is a challenge for big XML files in applications with critical performance requirements and requirements for low memory usage.

SAX on the other hand sequentially parses an XML file and invokes different call back methods of a default handler. Unlike DOM, SAX performs a one-time look-through into the XML file and does not create any in-memory structure of the XML data file.



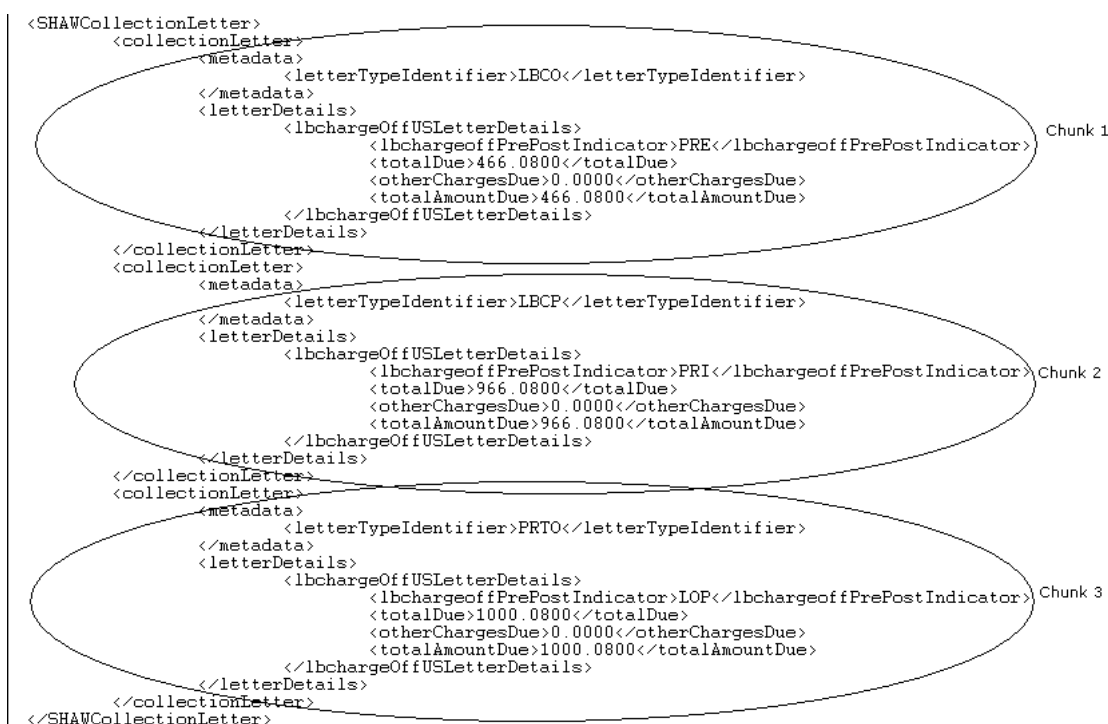
Here, SAX is used as a parsing methodology for splitting big sized XML data files into chunks of data files.

A default handler is used for implementing the call backs namely endDocument(), startElement(), endElement(), characters(). These call-backs are implemented such that the XML file is intelligently split into multiple chunks of files maintaining the original layout in the layout of the chunked files. *Chunking is performed at element level which has a recurring occurrence in the data file.*

The solution is exposed as an API called XML Splitter containing the interface and the implementation library.

4. How XML Splitter works

Following is an XML file whose layout is given in the diagram below:



Splitter will then start parsing the XML file using SAX methodology.



A default handler is embedded with the solution that implements the callbacks in such a way that XML content is appended in a buffer inside each callback. When the 10th element is reached during parsing the buffer is written to a file.

The buffer is then refreshed and parsing is continued in the same way till the end of file is reached and all files have been generated.

Each split file has a unique number appended to its file name that indicates the split file count.

5. Benefits and Constraints

Benefits:

- Can split big XML files safely into smaller and easy processing XML files maintaining the data structure and layout of the original file
- The smaller sized XML files containing data in chunks can be easily loaded into memory for reading and processing
- Following is the memory usage statistics for different file sizes

Size of XML File	Memory Consumed (MB) (SAX)	Memory Consumed (MB) (DOM)
1 MB	15	23
1.5 MB	16	25
2 MB	19.5	30

Constraints

- Current version of XML Splitter should not be used for XML files that has elements at the same level as the element at which the split occurs
- Needs to be thoroughly tested with different possible types of XML files having complex data structures and layouts